Universidad Rey Juan Carlos

Grado en Ingeniería en Telemática

Escuela Técnica Superior de Ingeniería Telecomunicación

Curso académico 2017/2018

Trabajo Fin de Grado

Robot submarino OpenROV y ROS

Autor: Saúl Ibáñez Cerro

Tutor: Francisco Martín Rico

A

Agradecimientos

Después de un intenso período, de meses duros por fallos de hardware y sistema que han conseguido llevarme hasta el límite, pero por fin puedo escribir este apartado de agradecimientos para finalizar mi trabajo de fin de grado.

Primero, quiero agradecer a mis padres todo el esfuerzo que han realizado por mi, por todos estos años de carrera apoyándome (y los años antes de empezar la carrera), muchas gracias por estar siempre ahí, por quererme y por animarme cuando más lo necesitaba. También quiero agradecer al enano de mi corazón, mi hermano, con el que siempre he podido hablar de mis quebraderos de cabeza. Por supuesto, también merece una mención especial mi pareja, Paula, no sé si hubiera podido con la presión y desánimos si no fuera por ella. Me ha ayudado más que nadie, en los momentos de un examen suspenso o una mala nota, siempre me ha apoyado y dado aliento, le debo muchísimo. Otro apoyo muy grande que he tenido han sido los padres de mi novia, que han estado siempre ahí cerca ayudándome. A mis abuelos por apoyarme y a todos los amigos y compañeros que me han ayudado durante toda esta etapa. Por supuesto, quiero mostrar mi gratitud a un profesor que me ha marcado. Por la forma de enseñar y mostrarnos su asignatura, el profesor de robótica, Francisco Martín, también por darme la oportunidad de hacer un TFG distinto.

De verdad, muchas gracias a todos, todos los éxitos que consiga en la vida, serán en parte gracias a vosotros.

Índice general

1. Introducción **1**

1.1. Robótica móvil 1

1.2. Robots Submarinos 2

1.3. ROS 4

1.4. Estructura de la memoria 4

2. Objetivos y Metodología **6**

2.1. Descripción del problema y requisitos 6

2.2. Objetivo del proyecto 6

2.3. Metodología de desarrollo 7

2.4. Plan de trabajo 8

3. Plataforma OpenROV **9**

3.1. Robot Submarino OpenROV 9

3.2. Estado del Arte 10

3.3. Estructura y Montaje 11

3.3.1. Estructura de OpenROV 11

3.3.2. Tapas del chásis 12

3.3.3. Electrónica 13

3.3.4. Enrutamiento 15

3.3.5. Final 15

3.4. Conectividad OpenROV 16

3.4.1. Diseño de Conectividad 16

3.4.2. Puesta a punto de los elementos 20

3.4.3. Web/Cockpit 23

3.4.4. Mantenimiento 28

3.4.5. Problemas 29

4. Integración de ROS en OpenROV **32**

4.1. Configuración del ROV 32

4.2. Configuración de ROS 36

4.3. Conexión de ROS en OpenROV 38

5. Pruebas de Campo **42**

6. Conclusiones **43**

Índice de figuras

1.1. Robots submarinos 4

2.1. Modelo en cascada. 7

3.1. OpenROV 10
3.2. Piezas de los chásis. 11
3.3. Chasis principal. 12
3.4. Soporte de la cámara y electrónica 12
3.5. Tapa inferior 13
3.6. Raspberry Pi + PLC + BeagleBone 14
3.7. Electrónica de la cámara 14
3.8. Soldadura de cables 16
3.9. PLC externo 17
3.10. BeagleBone configurada 18
3.11. Conexión Windows 8 18
3.12. Actualización firmware 20
3.13. Foco cámara 20
3.14. Láser 21
3.15. Motores 21
3.16. Tubo 22
3.17. Tapa superior 22
3.18. Cockpit (1) 23
3.19. Cockpit (2) 24
3.20. Cockpit (3) 24
3.21. Cockpit (4) 25
3.22. Cockpit (5) 25
3.23. Ajustes 26
3.24. Diagnóstico 27
3.25. Servicios 27

3.26. DB-25 30

4.1. Diagrama de la conexión ROS y OpenROV 33
4.2. SSH 33
4.3. Dependencias Roslibjs 34
4.4. Canvas 35
4.5. Roscore 39
4.6. Launch 39
4.7. Conexión en ROV 40
4.8. rostopic list 40
4.9. rostopic info (1) 41
4.10. rostopic info (2) 41

Capítulo 1

Introducción

En este capítulo se ubicará el proyecto en el contexto del robot submarino OpenROV y se enumeraran los distintos problemas que se presentan. Se expondrá el montaje y puesta a punto del OpenROV y la integración del framework ROS a la arquitectura del robot submarino. Por último se comentará la estructura de la memoria y se explicarán brevemente las partes por las que está formada.

1.1. Robótica móvil

Tradicionalmente las aplicaciones de la robótica estaban centradas en los sectores manufactureros más desarrollados para la producción masiva: industria del automóvil,transformaciones metálicas, etc.

A principios de los años sesenta se introducen en la industria, de modo significativo, los robots manipuladores como un elemento más del proceso productivo.

Los trabajos desarrollados por los robots manipuladores consistían frecuentemente en tareas repetitivas.

Una definición correcta de robot móvil plantea la capacidad de movimiento sobre entornos no estructurados, de los que se posee un conocimiento incierto, mediante la interpretación de la información suministrada a través de sus sensores y del estado actual del vehículo.

En general se considera que existen tres clases de robots:

- **Industriales** son los de mayor difusión en tareas de alcance económico, formados por una estructura mecánica articulada, que se mueve adoptando distintas configuraciones por las órdenes recibidas de un equipo de control basado normalmente en un microprocesador.

- **Medicos** de cooperación o de rehabilitación están concebidos como prótesis inteligentes para los disminuidos físicos. Se procura que tenga la estética correspondiente a la extremidad humana.

- **Móviles** son dispositivos de transporte automático, es decir, una plataforma mecánica dotada de un sistema de locomoción capaz de navegar a través de un determinado ambiente de trabajo.

  Los robots móviles operando en grandes ambientes no estructurados deben enfrentarse con significativas incertidumbres en la posición e identificación de objetos, la complejidad es tal que, trasladarse desde un punto A hasta un punto B es una actividad arriesgada para un robot móvil. El principal problema a resolver en un robot móvil es generar trayectorias y guiar su movimiento.

  El robot móvil autónomo se caracteriza por una conexión inteligente entre las operaciones de percepción y acción, que define su comportamiento y le permite llegar a la consecución de los objetivos programados sobre entornos con cierta incertidumbre.

  - **Percepción** La capacidad de percepción del robot móvil se traduce en la síntesis de toda la información provista por los sensores, con el objeto de generar mapas globales y locales del entorno de acuerdo a los diversos niveles de control.
  - **Razonamiento (acción)** El robot móvil debe ser capaz de decidir que acciones son requeridas en cada momento, según el estado del robot y el de su entorno, para alcanzar su(s) objetivo(s).

1.2. Robots Submarinos

Durante los últimos años, el uso de vehículos robots submarinos ha aumentado rápidamente, desde que tal vehículo se puede operar en las áreas más profundas y más arriesgadas que buzos no pueden alcanzar. Entre las aplicaciones potenciales de tales vehículos tenemos el área de la pesca, control submarino de contaminación, y la manipulsación y limpieza del océano, así como de sitios nucleares.

Se trata de un área de diversas aplicaciones en el mundo real, ya que permite y facilita la exploración, la búsqueda y rescate y estudios científicos en zonas muy profundas del mar, a pesar de problemas que se pueden presentar, como son la alta dinámica y la naturaleza del ruido del medio submarino, así como también la carencia de puntos de referencia y las limitaciones en las comunicaciones por efecto del agua.

Podriamos diferenciar las diferentes áreal potenciales en las cuales tienen una aplicación como:

- **Ciencia**

  Generación de mapas del suelo marino.

  Rápida respuesta a eventos eceanográficos y geotérmicos.

  Muestreo geológico

- **Militar**

  Búsqueda y eliminación de minas submarinas.

  Reconocimiento de rutas, áreas y zonas.

  Vigilancia de áreas de interés.

  Búsqueda y rescate en combate.

  Adquisición de objetos.

  Ajuste indirecto de armas de fuego.

  Seguridad en zonas cercanas.

  Soporte al desarrollo de la situación.

  Soporte a la preparación del campo de batalla.

  Soporte a la evaluación de daños en batalla.

- **Minería oceánica e industria del petróleo**

  Estudio del oceano y evaluación de recursos.

  Construcción y mantenimiento de estructuras submarinas.

- **Otras**

  Inspección en barcos del casco y los tanques internos.

  Inspección de plantes nucleares.

  Instalación e inspección de cables de comunicación y energía.

  Paseos turísticos bajo el mar.

  Etc.

(a) Robot Deep Trekker (b) Robot OpenROV

Figura 1.1: Robots submarinos

1.3. ROS

Sistema Operativo Robótico (en inglés *Robot Operating System, ROS*[1]) es un framework para el desarrollo de software para robots que provee la funcionalidad de un sistema operativo en un clúster heterogéneo.

ROS provee los servicios estándar de un sistema operativo tales como abstracción del hardware, control de dispositivos de bajo nivel, implementación de funcionalidad de uso común, paso de mensajes entre procesos y mantenimiento de paquetes. Está basado en una arquitectura de grafos donde el procesamiento toma lugar en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros.

ROS tiene dos partes básicas: la parte del sistema operativo, ros, como se ha descrito anteriormente y ros-pkg, una suite de paquetes aportados por la contribución de usuarios (organizados en conjuntos llamados pilas o en inglés stacks) que implementan la funcionalidades tales como localización y mapeo simultáneo, planificación, percepción, simulación, etc.

ROS es software libre bajo términos de licencia BSD. Esta licencia permite libertad para uso comercial e investigador. Las contribuciones de los paquetes en ros-pkg están bajo una gran variedad de licencias diferentes.

1.4. Estructura de la memoria

En este documento se describen los aspectos más relevantes del desarrollo y montaje del robot. La memoria esta dividida en 6 capítulos. El primer capítulo, Introducción, se

[1] http://wiki.ros.org/

ha realizado una presentación de los robot móviles, en particular, del robot submarino OpenROV. En el segundo capítulo se define el problema y se establecen los objetivos. El montaje del robot se especifica en el tercer capítulo. En el siguiente capítulo, el cuarto, se explicará el entorno y las herramientas utilizadas. En el quinto capítulo, se mostrará la integración del OpenROV con ROS y en el último, el sexto, habrá ejemplos de como ha funcionado el robot en un entorno real.

Capítulo 2

Objetivos y Metodología

Tras haber presentado el contexto en el que se desarrollará este proyecto, en este capítulo se fijarán los objetivos y se expondrán los requisitos para llegar a la solución.

2.1. Descripción del problema y requisitos

El fondo oceánico es uno de los grandes misterios de la actualidad. La mayor parte del suelo oceánico no ha sido explorado, y eso supone un gran problema. Con este robot, podremos rozar un poco este mundo submarino tan fascinante, e intentar descubrir algunas de las magníficas maravillas que esconde.

El robot contiene una cámara de alta visualización que permite realizar un análisis exhaustivo, y para entornos más oscuros precisaremos de las luces y los láseres incorporados, con ellos, podremos visualizar mejor el entorno y saber la distancia a la que se encuentren los objetos o animales.

Los requisitos indispensables para realizar el proyecto son:

1. **Robot móvil OpenROV 2.8.**

2. **BeagleBone.** La placa en la que se instalará el software de OpenROV.

3. **Herramientas de soldadura, montaje y sujeción.**

4. **Portatil.** Ejecutará el código para manejar el entorno del robot.

2.2. Objetivo del proyecto

En este proyecto, se quiere acercar más al mundo el robot submarino OpenROV. La exploración del mundo submarino. Se construirá de cero el robot submarino, utilizando los materiales dados por la comunidad de OpenROV. A la placa de OpenROV se le instalará un PLC para la comunicación del robot con el portátil y una BeagleBone, que

llevará implantado el software necesario para la comunicación y funcionamiento con el robot. Además, se implantará una biblioteca de ROS dentro de OpenROV para que también se pueda comunicar a través del framework de robótica más famoso. Con esta biblioteca instalada y ejecutando el ROS Master en el portátil, se podrá realizar una comunicación.

2.3. Metodología de desarrollo

En el desarrollo del sistema descrito, el modelo de ciclo de vida utilizado ha sido el modelo Modelo en cascada.

Es un proceso secuencial, fácil de desarrollo en el que los pasos de desarrollo son vistos hacia abajo (como en una cascada de agua) a través de las fases de análisis de las necesidades, el diseño, implantación, pruebas (validación), la integración, y mantenimiento.

Los principios básicos del modelo de cascada son los siguientes:

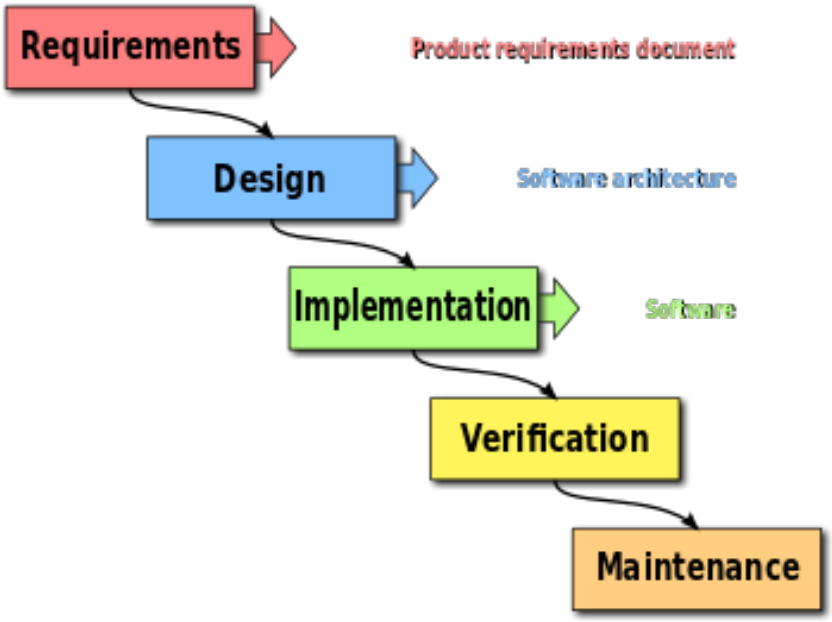


Figura 2.1: Modelo en cascada.

- El proyecto está dividido en fases secuenciales, con cierta superposición y splashback aceptable entre fases.

- Se hace hincapié en la planificación, los horarios, fechas, presupuestos y ejecución de todo un sistema de una sola vez.

- Un estricto control se mantiene durante la vida del proyecto a través de la utilización de una amplia documentación escrita, así como a través de comentarios y aprobación.

2.4. Plan de trabajo

Para poder abordar el problema se han marcado una serie de objetivos a completar. Dichos hitos son los siguientes:

1. Estudio, comprensión y montaje del OpenROV 2.8.
2. Una vez terminado el montaje, se realizará la conexión del ROV para comunicarse con el portátil.
3. **Fase de pruebas**, se probará el robot en distintos entornos para comprobar su funcionamiento.
4. Estudio y comprensión de los paquetes de ROS roslibjs[1] y rosbridge[2]
5. Integración de ROS y OpenROV.

[1] http://wiki.ros.org/roslibjs
[2] http://wiki.ros.org/rosbridge_suite

Capítulo 3

Plataforma OpenROV

3.1. Robot Submarino OpenROV

La robótica submarina lleva desarrollándose desde mediados de los años sesenta, aunque fuera una realidad desconocida para el gran público. Los robots submarinos costaban millones y tenían el tamaño de un todoterreno. Un robot con una cámara que se retransmite en directo a una página web. Su principal virtud es que OpenROV[1] funciona con código abierto.

En general, sus características entran dentro de los parámetros presentados a continuación:

- **Cámara web de alta definición para video en vivo HD** Se transmite vídeo de alta definición a un portátil a través de un cable trenzado de dos hilos.

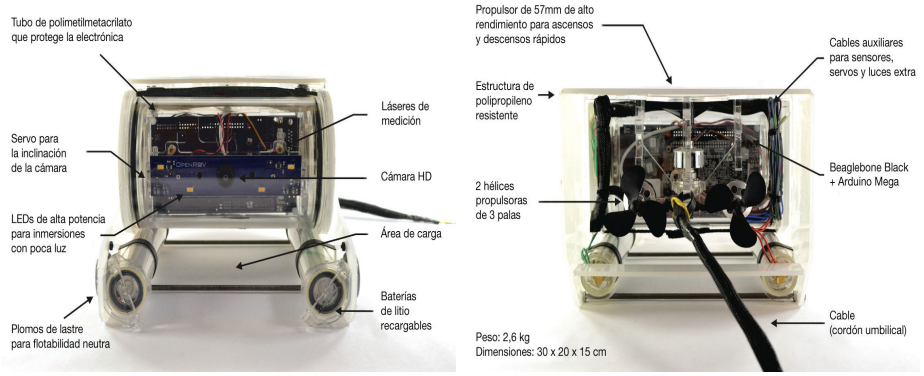
- **Tres servomotores** Se utilizan para la movilidad del dispositivo. Son de muy bajo mantenimiento y puedan estar en contacto directamente con el agua.

- **Flotabilidad neutra y altamente maniobrable** El diseño compacto y ligero permite una alta maniobrabilidad utilizando un diseño de tres actuadores. Se puede sumergir a una profundidad de 100 metros.

- **Interfaz web fácil de usar** En ella se puede manipular la cámara, los láser, focos y motores. No necesita conexión a Internet, ni hay necesidad de instalar un software.

- **Soporte de la comunidad de OpenROV** Se tiene soporte de la amplia comunidad que da soporte al proyecto OpenROV.

[1] https://www.openrov.com/



(a) OpenROV parte frontal (b) OpenROV parte trasera

Figura 3.1: OpenROV

3.2. Estado del Arte

El mundo de los robots submarinos ROV (*Remote Operated Vehicle* o vehículo de operación remota) sigue creciendo año tras año y no debe sorprender. El 70 % por ciento de la superficie de la Tierra está cubierta de océanos. El 95 % del fondo de los océanos sigue sin mapear. De hecho, se conoce más sobre la superficie de la Luna que de las profundidades oceánicas. Tanto es así, que doce hombres han puesto pie en la Luna, pero sólo dos lo hicieron en la Fosa de las Marianas, la parte más profunda de nuestros mares, con aproximadamente 11 kilómetros de profundidad.

La inherente necesidad del ser humano de ir hacia lo desconocido para expandir nuestras fronteras no tiene límite. Esto se ve reflejado en el aumento de proyectos que buscan expandir estas fronteras submarinas.

En base al estudio del estado del arte, se establecieron las variables físicas necesarias para poder manipular el robot en cuestión, estas variables son las siguientes:

- **Aceleraciones en X, Y, Z** Permite obtener el ángulo con respecto al plano. Por odometría se puede obtener velocidad y posición estimadas.

- **Velocidad angular en p, q y r (roll, pitch y yaw)** Por medio de odometría, permite estimar perturbaciones en la rotación del robot.

- **Temperatura Externa** Puede utilizarse para hacer un registro de datos que permita caracterizar unmedio marino.

- **Temperatura Interna** Puede proveer de alerta de sobrecalentamiento de los sistemas a bordo.

- **Velocidades Lineales** En conjunto con la estimación de velocidad por odometría, puede utilizarse para compensar errores acumulativos.

- **Orientación con respecto al norte magnético** Permite obtener una referencia para mantener una referencia con respecto a la tierra.

- **Presión Relativa** De manera indirecta, permite conocer la profundidad.

- **Carga de Baterías** Permite conocer el estado de carga de las baterías. En caso de estar en un nivel crítico, se pueden ejecutar modos de bajo consumo o de recuperación de la plataforma.

3.3. Estructura y Montaje

3.3.1. Estructura de OpenROV

Antes de comenzar con el montaje de la estructura, la cámara y el chasis del ROV, se necesitarán ciertas herramientas para el correcto ensamblaje.

Las herramientas necesarias son: guantes, gafas, cúter, cemento acrílico y pegamento. Una vez hayamos obtenido todos los materiales empezaremos con el montaje.

Figura 3.2: Piezas de los chásis.

Se separan las piezas para realizar su correcto montaje. Una vez localizadas las piezas para la estructura, se le quitará el papel protector a la pieza, lo cual la dejará de un color trasparente. Iremos pegando y dejando secar para que la composición sea la correcta. Al final, quedará una estructura como la de la imagen.

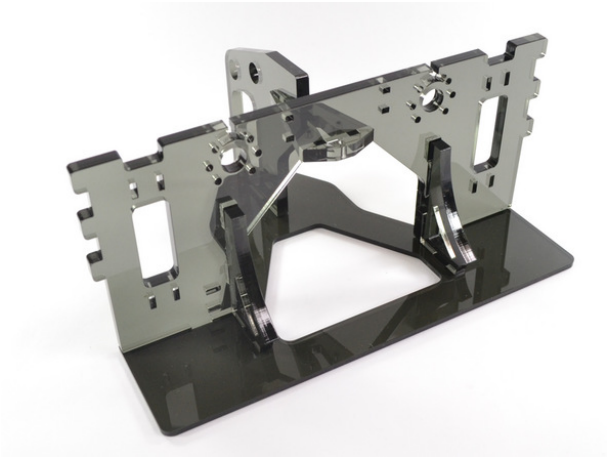
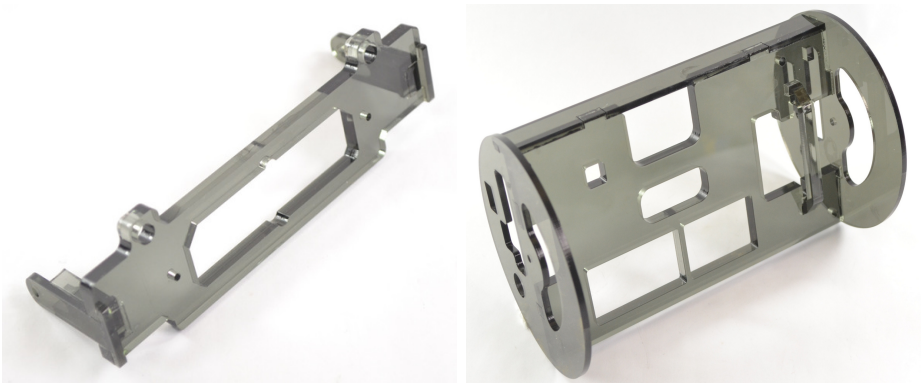


Figura 3.3: Chasis principal.

Para el montaje de la estructura del soporte de la cámara y el chasis de la electrónica se realizará el mismo procedimiento que en el caso anterior, con un resultado como el de las siguientes fotos:



(a) Soporte de la cámara (b) Chasis de la electrónica

Figura 3.4: Soporte de la cámara y electrónica

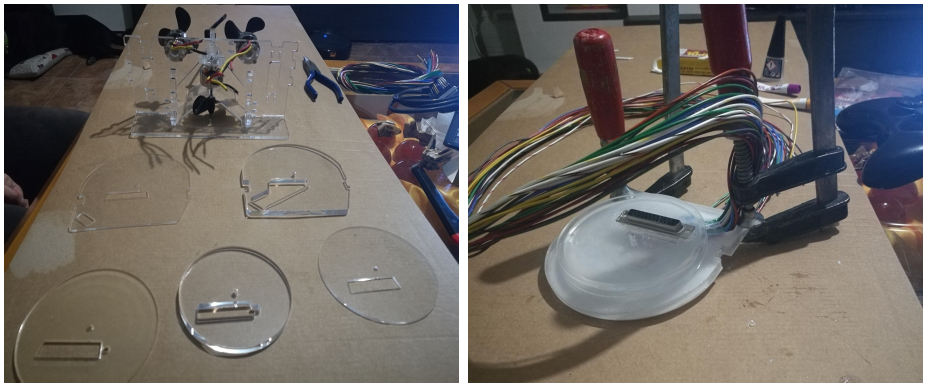
3.3.2. Tapas del chásis

Al igual que en el punto anterior, antes de comenzar con el montaje de la tapa, se necesitarán ciertas herramientas para el correcto ensamblaje.

Las herramientas necesarias son: guantes, gafas, cúter, cemento acrílico, SuperGlue y epoxy (epoxy es un tipo de pegamento que soporta estar en contacto con el agua). Una vez hayamos obtenido todos los materiales empezaremos con el montaje.

En este caso, existen dos tapas. En la primera (la superior), necesitaremos una jeringa proporcionada por la comunidad de OpenROV, la cual utilizaremos para preparar la impermeabilización y el sellado de los componentes software del robot.

En la segunda tapa, está el conector DB-25, el cual, conecta todos los componentes eléctricos a la placa del controlador. Una vez realizado el montaje de la tapa, se pegará en el espacio sobrante el conector DB-25 y se colocarán los cables adecuadamente en el hueco de la tapa. Cuando esté todo debidamente colocado se pegará la tapa superior al montaje realizado, lo cual dejará todo debidamente sellado. Este hueco, se rellenará con el epoxy para que no entre agua por la ranura.



(a) Tapa inferior desmontada (b) Tapa inferior montada

Figura 3.5: Tapa inferior

3.3.3. Electrónica

Este es el punto crucial del funcionamiento del robot y se necesita especial atención en cada uno de los pasos.

Al igual que en los apartados anteriores, necesitaremos material especial, entre los que están el soldador, estaño, PLC, BeagleBone, tijeras y destornilladores.

Para empezar, desmontaremos los PLC para sacar el hardware, el cual utilizaremos más adelante. Uno de los PLC estará interno en el robot, y el otro, será el extremo conector que comunicará el ROV con el portátil.

Continuaremos con el BeagleBone Black (BBB), el cual, es el procesador principal en el ROV. Funciona junto con un Ardunio en el tablero del controlador para controlar las muchas funciones del ROV.

El cable USB del BeagleBone lo utilizaremos más adelante.

Una vez juntos el PLC, el BeagleBone y la placa del ROV, uniremos estos tres elementos para formar el hardware del robot. Se conectará primero la placa del ROV,

en la que el conector RJ-45 irá conectado al hardware PLC. Una vez realizada esa conexión, procederemos a implantar en las ranuras el BeagleBone, y atornillaremos para obtener una mejor sujeción.

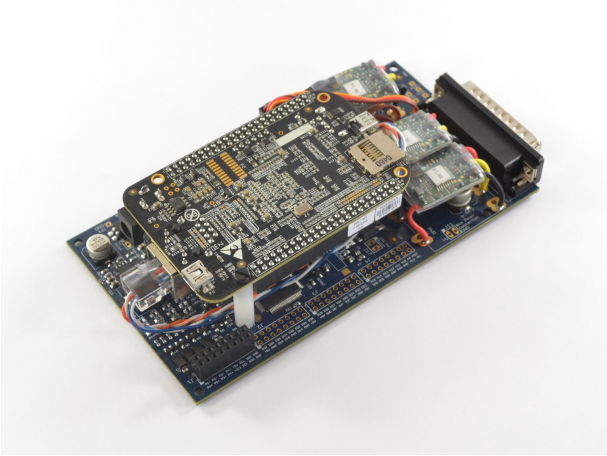


Figura 3.6: Raspberry Pi + PLC + BeagleBone

Después, conectaremos el servo al chasis, el cual va a servir para mover la cámara verticalmente. Hay que asegurarse de que quede centrado.

Continuaremos con la soldadura de los láser y unos cables a la placa de video del ROV.

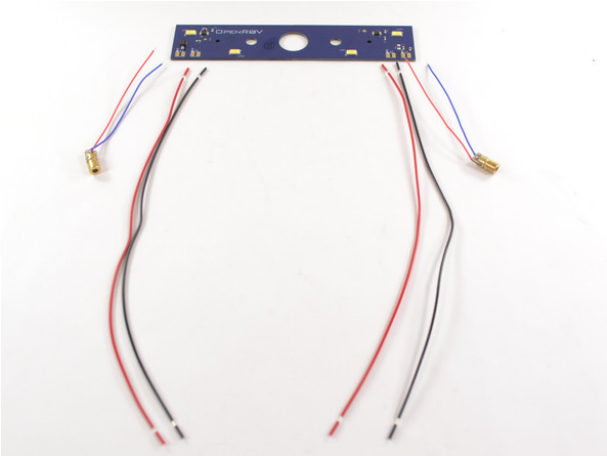


Figura 3.7: Electrónica de la cámara

Cuando estén debidamente soldados, se colocará el chasis de la cámara, junto con lo que acabamos de soldar y la cámara. A estos tres elementos los juntaremos con tornillos y tuercas. Para finalizar con este paso, comprobamos que hay un conector al cual enchufaremos un cable que irá directo a la placa principal del ROV.

Para la sujeción de la cámara utilizaremos el chasis principal, al que atornillaremos para fijar la cámara. En la parte izquierda del chasis principal, existe un hueco en el que irá el motor servo, el cual ha sido modificado para esa ubicación, eliminándole una de las astas para que se quede fijo en una posición y pueda mover la cámara en el eje vertical.

Para finalizar este apartado, se conectará el hardware del robot en el lado opuesto de la cámara, y se fijará con tornillos. Se realizará la conexión de los cables a sus ranuras correspondientes.

3.3.4. Enrutamiento

En esta fase, se va a requerir unas herramientas más específicas para realizar el montaje, ya que necesitaremos soldar e impermeabilizar con una pistola de calor, además, de tijeras, epoxy, super Glue y bridas.

Empezaremos calculando la distancia de los cables y separaremos los de color verde claro y naranja, para la conexión de las baterías. Los cables verdes oscuros, rojos y azules, los utilizaremos para el montaje de los servomotores. Iremos fijando los cables con las bridas para una correcta colocación.

Cada uno de los tres motores servo, lleva tres cables, los cuales soldaremos con los cables correspondientes. Una vez soldados, con una silicona especial se rellenará el espacio del cable soldado y con la pistola de calor se fijará e impermeabilizará.

Los diez cables sobrantes se sellarán, ya que servirían para sensores y luces extra que no utilizaremos en este proyecto.

Continuaremos con el montaje de los dos tubos de la batería. Primero, necesitaremos realizar el montaje de las piezas y soldaremos el muelle (borne) con un chip facilitado por la comunidad de ROV. Esta parte irá con el polo negativo de las baterías. En el extremo del cable sobrante, se rellenará el terminal con estaño y se soldará al cable formando la parte positiva en la que se conectará la batería.

3.3.5. Final

En la parte final del montaje, será necesaria una soldadura con el cable de par trenzado que es el encargado de mandar la señal al portátil.

Para asegurar esa soldadura, utilizaremos una brida de un tamaño mayor a las usadas anteriormente, y soldaremos un cable preservado en el apartado anterior con el par trenzado.

Este cable de par trenzado mide cien metros, el cual, en el extremo opuesto al ROV, irá conectado al PLC. Gracias a este cable, el robot se podrá sumergir a una distancia

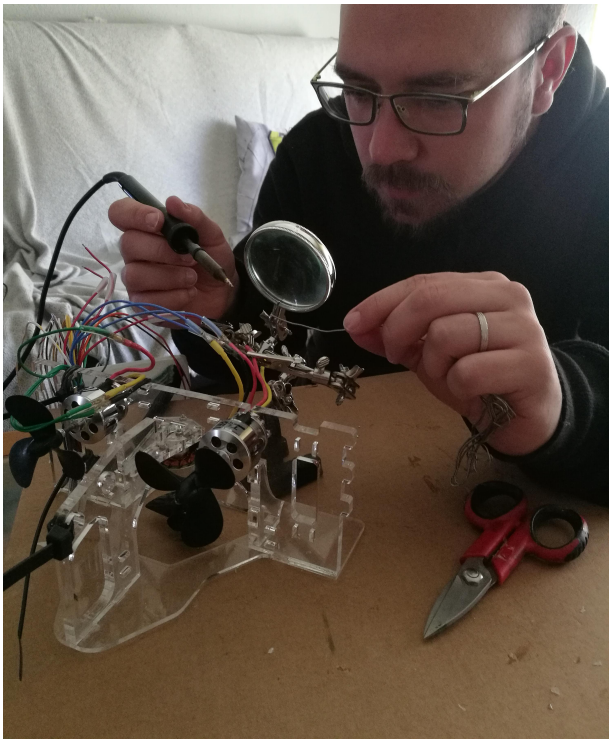


Figura 3.8: Soldadura de cables

de cien metros. Por eso, y para mayor seguridad, usaremos la brida, enroscaremos el cable en ella y lo finalizaremos recubriéndolo con cinta adhesiva. En el cuerpo principal del ROV, sujetaremos con unas gomas los tubos que llevarán las baterías, y lo fijaremos en los extremos con dos varillas de metal.

Aislaremos el hardware con un tubo cilíndrico trasparente de polimetilmetacrilato, pero antes de fijarlo, limaremos los extremos del cilindro para una mejor conexión con las tapas.

Cuando todo el montaje esté terminado, pondremos más seguridad al hardware, añadiendo una cuerda que hará presión.

Además, colocaremos dos pesas de trescientos gramos en los brazos del ROV para que la sumersión sea más fácil y fiable.

3.4. Conectividad OpenROV

3.4.1. Diseño de Conectividad

Para encender el ROV, necesitaremos el cable mini USB y el cable ethernet debidamente colocado en el extremo del PLC y estos a su vez al portátil.

El conector USB actúa como el interruptor de encendido/apagado para el ROV.

Figura 3.9: PLC externo

El ROV tiene una dirección IP estática incorporada de 192.168.254.1, por lo que para conectarse con ella, la dirección IP de ethernet de su computadora debe estar en la misma subred, es decir, 192.168.254.2. La máscara de subred debe establecerse en 255.255.255.0.

Pero para poder mostrar la página, antes debemos realizar una serie de pasos, en la que necesitaremos una microSD para realizar la configuración del ROV.

- El primer paso, es descargar la imagen más reciente de OpenROV (30.0.3), una vez descargado, se descomprime el archivo .zip, dejando en el directorio de descarga el fichero .img.

- Para cerciorarse de que la mircoSD esta vacía, se formatea con el programa SDFormatter, luego, con el programa Win32DiskImager, escribiremos en la tarjeta microSD el fichero .img, esperamos alrededor de 5 minutos a que se complete la escritura. Cuando haya terminado, removemos la tarjeta microSD del portátil.

- Conectamos la tarjeta microSD a el Beagle Bone, y conectamos el ROV con el cable USB para encenderlo. Los LED de Usuario de la placa Beagle Bone parpadearán para mostrar que laa imagen se está aplicando a la memoria de dicha placa. El proceso dura aproximadamente 15 minutos, en ese tiempo, los cuatro led deben estar encendidos, de no ser así, habría que repetir este punto nuevamente.

  Cuando esté completo, retiraremos el cable USB del portátil para desconectar el ROV y retiraremos la tarjeta microSD de la placa Beagle Bone.

  Terminado este proceso, volveremos a conectar la tarjeta en el portátil.

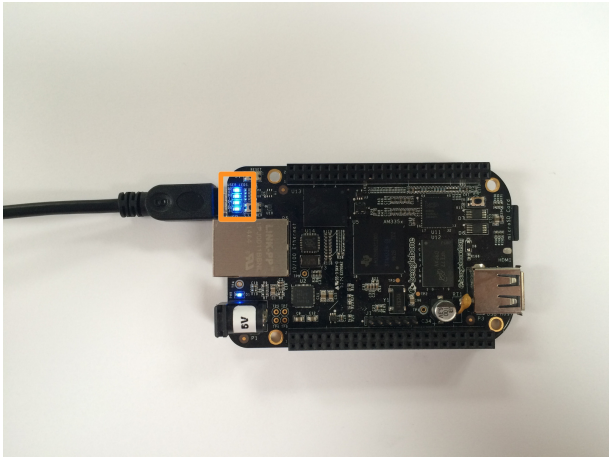


Figura 3.10: BeagleBone configurada

- Como se mencionó al principio del apartado, el ROV tiene una dirección IP estática incorporada de 192.168.254.1, por lo que para conectarse con ella, el ordenador debe tener una dirección similar pero con el último número configurado distinto de 1, por ejemplo, podemos usar la dirección "192.168.254.2". La máscara de subred debe establecerse en 255.255.255.0.

  En Windows 8 la configuración sería:

Configuración 3.1: Windows 8

```
Ir  al Panel de control.
Seleccionar Red e Internet.
Red y centro compartido  (click) cambiar configuracin de adaptador
   (click) ethernet.
Propiedades  (click) usar la siguiente direccin IP
Insertarla direccin IP 192.168.254.2
```

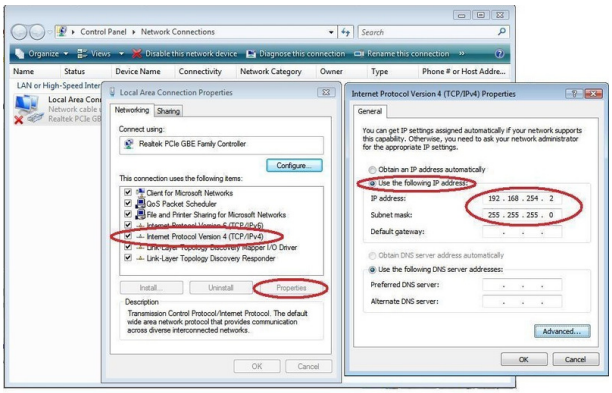


Figura 3.11: Conexión Windows 8

En Ubuntu la configuración de una nueva IP estática:

Configuración 3.2: Ubuntu

```
# Abrir el terminal (Ctrl + Atl + T).
# Comprobamos las interfaces del equipo:
$ ifconfig -a
# Editamos el archivo de configuracion de las interfaces:
$ sudo vim /etc/network/interfaces

# Configuracion de direccion IP fija para el interfaz eth0
  auto eth0
  iface eth0 inet static
  address 192.168.254.2
  netmask 255.255.255.0
  network 192.168.254.0
  broadcast 192.168.254.255
  gateway 192.168.254.1
```

Donde:

address es la dirección IP que quieres ponerle a tu máquina.

netmask es la máscara de subred de esa dirección IP.

network es la red a la que pertenece esa dirección IP.

broadcast es la direción IP de difusión de esa red.

gateway es la dirección IP de la puerta de enlace predeterminada.

Reiniciamos las interfaces de red para aplicar los cambios:

3.3: Reinicio

```
$ sudo /etc/init.d/networking restart
```

- Abriremos el navegador Google-Chrome, y en la barra de direcciones escribiremos la IP 192.168.254.1:8080, que es la dirección IP de OpenROV. Presionaremos ¡enter¿y en un periodo de aproximadamente 20 segundos aparecerá la interfaz de OpenROV.

- Para finalizar, deberemos actualizar el firmware, en la web, presionaremos el botón de configuración en la parte superior derecha de la pantalla y presionaremos "Cargar firmware desde la tarjeta SD a Ardruino" La tarjeta no microSD no debe estar en el ROV, presionaremos "Mostrar Detalles" y seguidamente, "Aplicar nuevo Firmware". El proceso dura aproximadamente entre 5-10 minutos. Una vez que la carga sea exitosa, se desconectará el cable USB del portátil para apagar el ROV, para realizar el reinicio, volveremos a conectar el cable USB al portátil.

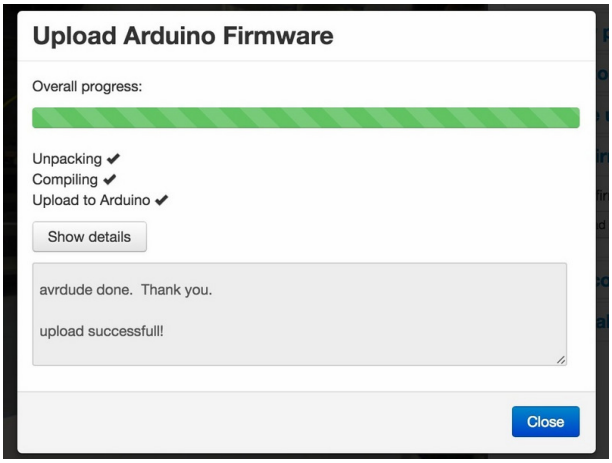


Figura 3.12: Actualización firmware

3.4.2. Puesta a punto de los elementos

Cámara

Ajustaremos el foco de la lente de la cámara hasta que la imagen pueda verse con total nitidez.

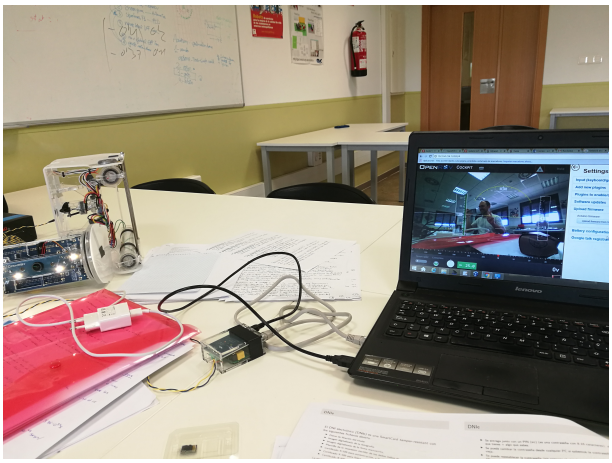
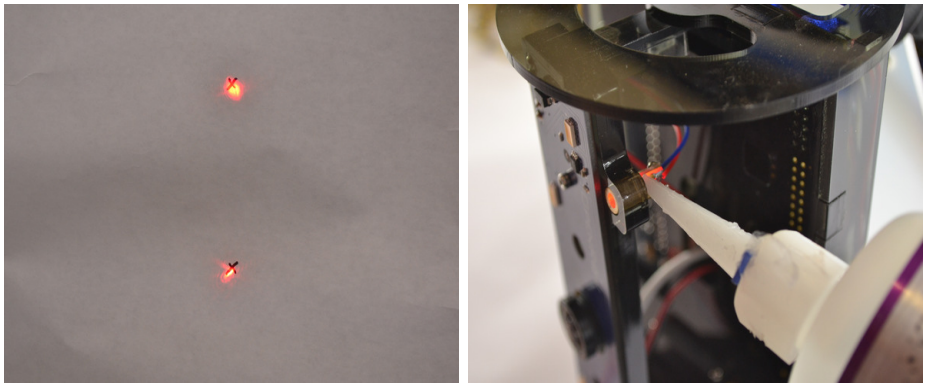


Figura 3.13: Foco cámara

Láser

El ROV llega incorporado dos láser, los cuales tendremos que montar en el chasis de la cámara. Anteriormente estos dos láseres ya han sido soldados en la placa de la cámara, con lo que el proceso que llevaremos a cabo será su calibración y pegado. Los puntos siempre estarán a diez centímetros de distancia uno del otro, actuando como una escala para mediciones subacuáticas. En un papel pondremos dos “X” separados entre si diez centímetros, y colocaremos la hoja de calibración entre tres y cuatro

metros de distancia de los láseres. El ROV y el folio deben colocarse a la misma altura. Presionamos la tecla L en la web de OpenROV para encender los láseres (si volvemos a presionar esa tecla, los láseres serán desactivados). Una vez este todo calibrado, pegaremos con Super Glue los láseres al chasis de la cámara.



(a) Distancia de los laseres (b) Fijado del los laseres

Figura 3.14: Láser

Motores

En este punto, se comprobará si las hélices giran en la dirección correcta. En la web, presionaremos la tecla de "flecha arriba" mientras miramos en que dirección giran los motores. El motor izquierdo (amarillo) debe funcionar en sentido contrario a las agujas del reloj. El motor derecho (azul) debe funcionar en sentido a las agujas del reloj.

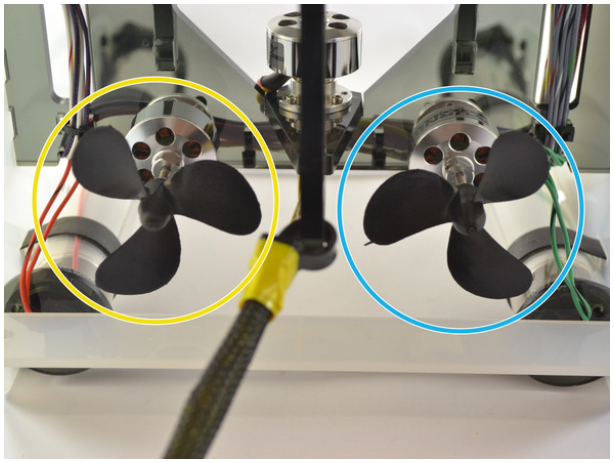


Figura 3.15: Motores

Si el giro fuera incorrecto, podemos invertirlo en la parte de "diagnóstico" de la esquina superior derecha del Cockpit. Seleccionamos la opción "inversa" del motor que queramos modificar.

Tubo cilíndrico trasparente de polimetilmetacrilato

El tubo cilíndrico contendrá todo el sistema hardware y software del ROV. Lijaremos los bordes interiores del tubo principal utilizando un papel de lija de grano mediano para que la junta se ajuste en el tubo. Limpiamos todo el polvo sobrante del proceso de lijado.

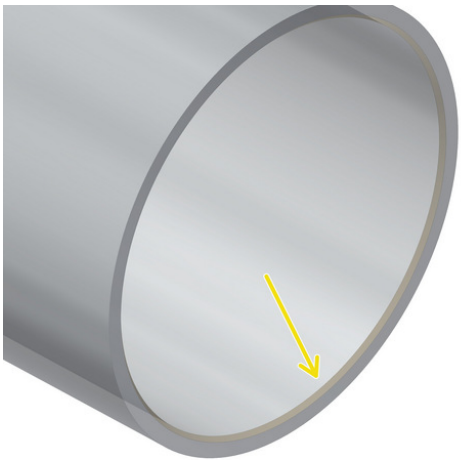


Figura 3.16: Tubo

Tapa superior

Las juntas son muy importantes para garantizar que los componentes electrónicos permanezcan secos. Después de instalar el tubo, nos aseguramos de que la junta se enganche a lo largo de todo el perímetro interior del tubo. El tubo electrónico necesita el orificio de ventilación proporcionado por el émbolo para igualar la presión cuando se colocan las tapas.

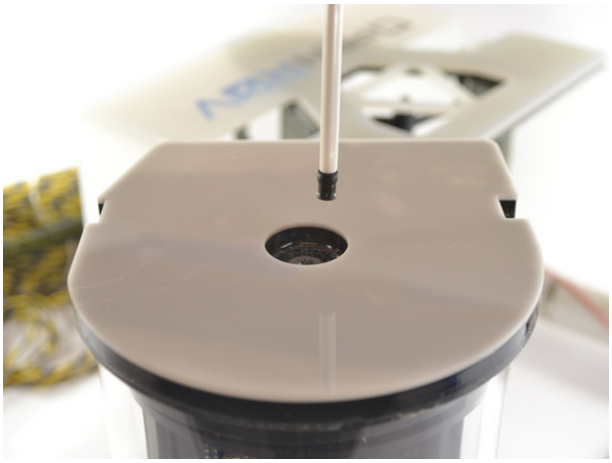


Figura 3.17: Tapa superior

3.4.3. Web/Cockpit

El ROV se maneja con Cockpit que es la interfaz de usuario y los sistemas de control para cualquier vehículo o dispositivo operado a distancia.
Funciona con Linux incorporado y microcontroladores para proporcionar el control tele-robot.

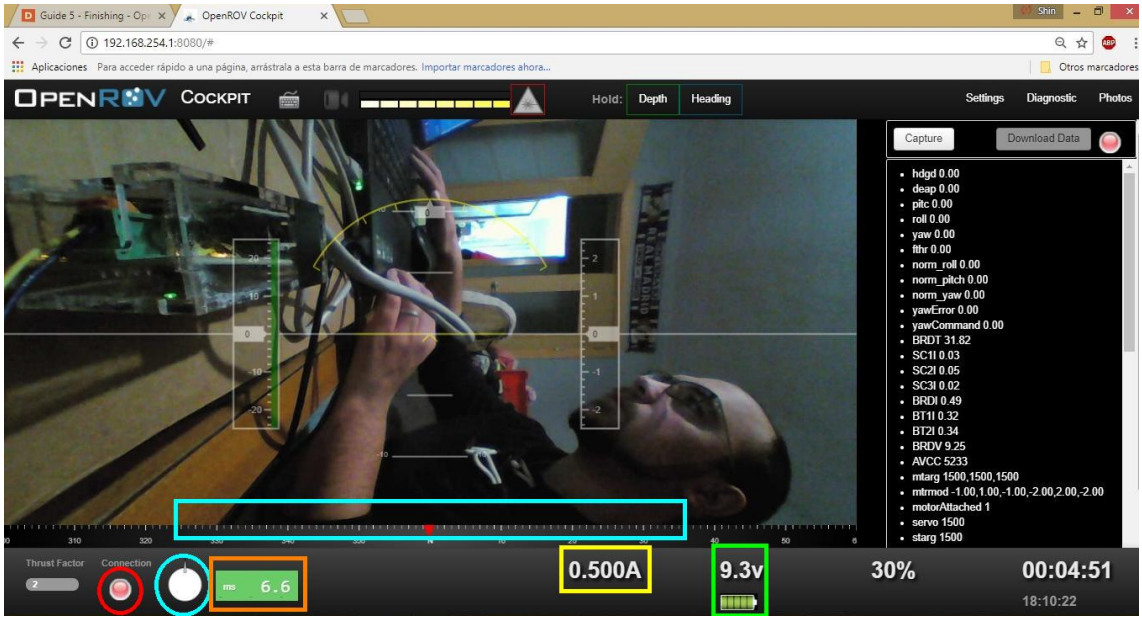


Figura 3.18: Cockpit (1)

- Conectividad (verde = conectado)
- Latencia (si hubiera algún problema mostraría +999)
- Consumo de corriente (solo los componentes electrónicos, sin incluir motores)
- Voltaje de la batería (para las baterías LiFe-PO4 nominales es alrededor de 9.2V, el ícono puede mostrarse medio lleno, lo cual es normal)
- Brújula/rumbo (solo con IMU (Unidad de medición inercial))

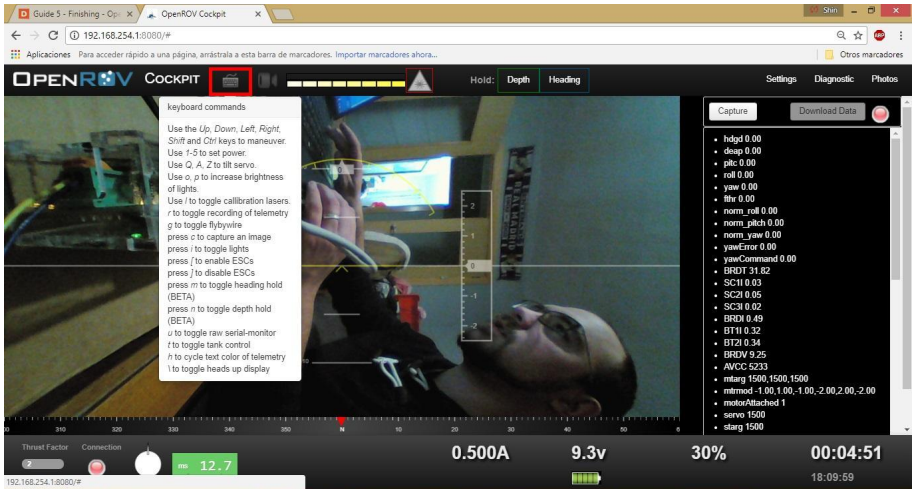


Figura 3.19: Cockpit (2)

- Pulsamos el botón señalado en rojo (el icono del teclado) para obtener los comandos de uso del robot.

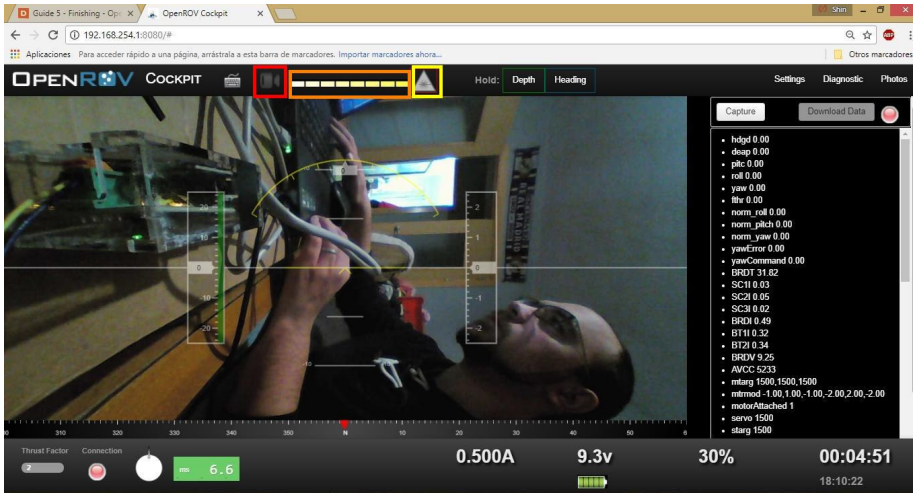


Figura 3.20: Cockpit (3)

- Indicador de la cámara
- Indicador de nivel de luz LED
- Indicador de encendido/apagado del láser

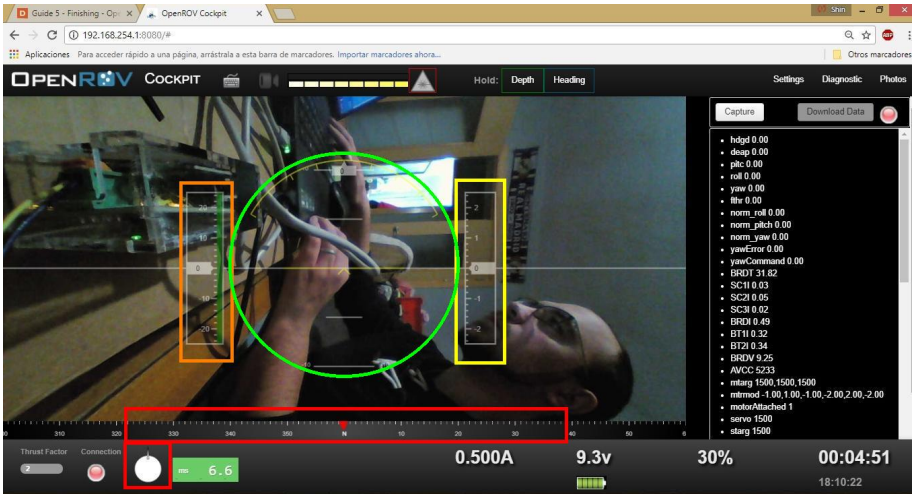


Figura 3.21: Cockpit (4)

- Rumbo (brújula). Esto se basa en la dirección que obtiene OpenROV al inicio. Se puede calibrar alineando el ROV al norte al inicio.
- Valor de empuje del motor (IMU no es necesario)
- Profundidad
- Horizonte artificial

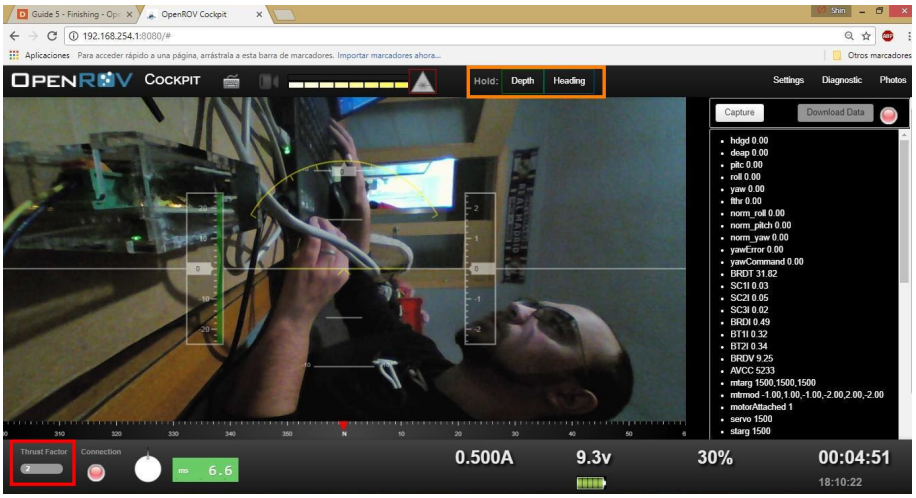


Figura 3.22: Cockpit (5)

- La configuración de empuje (1, la más baja y 5, la más alta) cambia la potencia otorgada a los motores. El uso típico es 2-3. El uso intensivo de nivel 5 agotará las baterías rápidamente y solo se recomienda para retiros rápidos y uso intermitente.

- En la parte de arriba se pueden ver los indicadores de profundidad y rumbo. Estos se alternan entre activar/desactivar para mantener la profundidad actual y/o el rumbo actual. Esta función solo funciona con la actualización profundidad/rumbo/IMU.

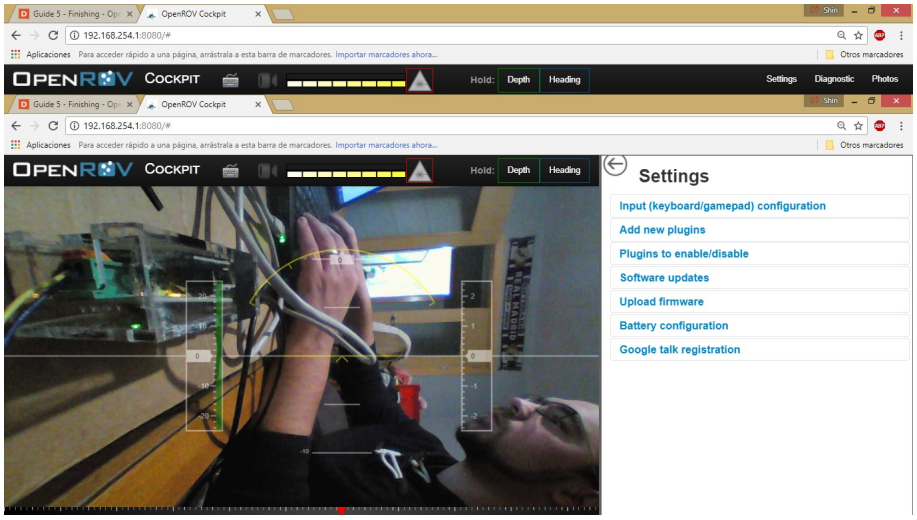


Figura 3.23: Ajustes

Panel de configuración:

- Configuración de entrada: aquí puede cambiar y personalizar las combinaciones de teclas y botones.
- Agregar complementos nuevos: esto es para usuarios que desean usar complementos de 3ra parte (avanzados).
- Complementos para habilitar/deshabilitar: activa y desactiva los complementos cargados en el punto anterior (avanzado).
- Actualización de software: se recomendamos que siempre actualice su software. Esto también le muestra la versión de software actual.
- Configuración de la batería: en este punto es donde establece los voltajes mínimo y máximo de la batería.
- Registro de Google Talk: esta pestaña permite acceder a las funciones de redes sociales del cockpit (avanzado).

Panel de diagnósticos:

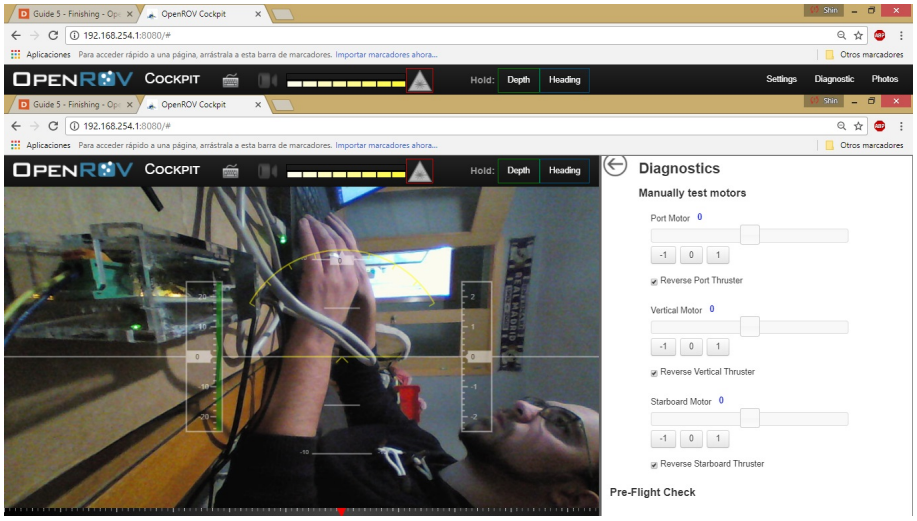


Figura 3.24: Diagnóstico

- Probar manualmente los motores, esto se usa para hacer funcionar los motores manualmente. Este panel también sirve para invertir la dirección de un cierto motor.

- Pre-Flight Check es un marcador de posición por el momento.

- La calibración para la brújula y la profundidad cero también se encuentran en este panel.

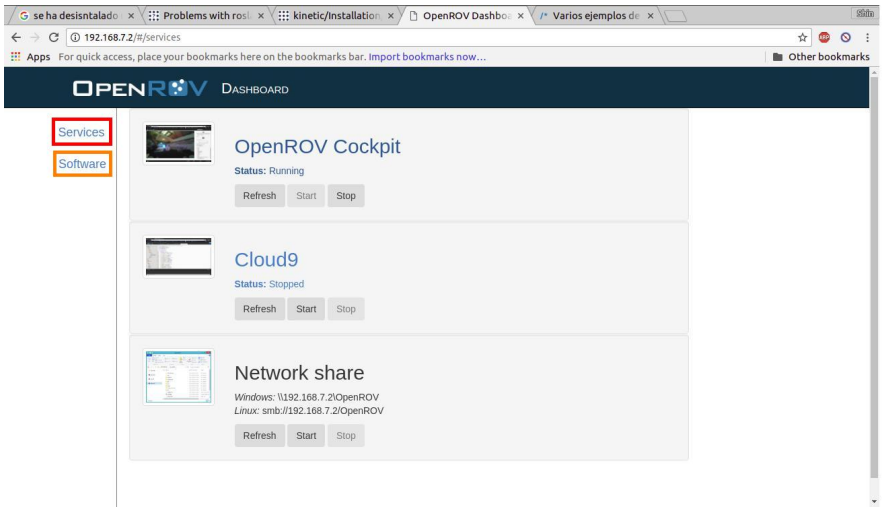


Figura 3.25: Servicios

El panel de configuración de OpenROV se encuentra en: 192.168.254.1

Hay dos partes:

- **Servicios**: puede reiniciar la cabina, ejecutar Cloud9 para editar el código e iniciar el recurso compartido de red para acceder a los archivos en el ROV.
- **Software**: puede determinar la versión de su software así como la actualización automática cuando esté conectado a Internet (software 30.0.3 y superior).

3.4.4. Mantenimiento

Enjuagar

Una vez que hayamos utilizado el robot, debemos rociarlo en agua dulce después de cada inmersión, ya que es una excelente manera de mantenerlo limpio y libre de desechos y minerales.
Cuanto más limpia sea el agua, mejor. En orden ascendente sería:

- El agua desionizada como mejor opción.
- El agua destilada.
- El agua dulce del grifo en tercer lugar.

Mantenimiento de los motores

La parte más importante del mantenimiento posterior a la inmersión es limpiar los motores. Los imanes expuestos tienden a corroerse fácilmente. Los rodamientos también deben lubricarse y protegerse del óxido.

- Primero, enjuagamos los motores con agua fría.
- Secaremos bien los motores con aire comprimido (si está disponible).
- Inmediatamente después, volveremos a aplicar la lubricación con spray de silicona. Moveremos los motores con las manos durante algunas rotaciones para asegurarse de que los quede todo bien lubricados. Esto debe hacerse antes y después de cada inmersión.

Secado

Debemos permitir que el OpenROV se seque completamente antes de guardarlo. Usar aire comprimido o enlatado puede ayudar.
La apertura de los tubos de la batería y el tubo principal permitirá que el aire circule y secará la humedad que pueda haberse acumulado durante la inmersión.

Carga de baterías

Mantener las baterías en grupos de seis permitirá evitar problemas. Cargaremos los seis a la vez, y se usará todo el grupo por cada inmersión.
Si alguna de las tres baterías en un tubo tiene un rendimiento bajo, se puede dañar las células. Para evitar esto, podemos comprobar las baterías con un polímetro después de que se hayan cargado para asegurarse de que se carguen completamente.
Esto también permitirá saber cuándo es el momento de reemplazarlos, ya que eventualmente perderán su capacidad de cargar y descargar.
Mantener las baterías secas siempre que sea posible y sin corrosión.

3.4.5. Problemas

Vaho

Que el tubo principal se empañe es un problema común.

Algunas veces esta es la indicación de una fuga. Si se nota que el vaho aparece repentinamente o durante una inmersión, lo mejor será salir a la superficie y verificar si hay fugas.

El vaho se puede solucionar colocando una nueva bolsa de desecante en el tubo principal unas 1-2 horas antes de la inmersión.

Los productos químicos anti-vaho también pueden ayudar. Estos están disponibles en cualquier tienda de buceo.

Video

Muchos problemas de video pueden ocurrir durante su inmersión.

El más común es un corte o .ªpagón" del video. Esto se puede solucionar actualizando periódicamente la imagen del software en el BeagleBone.

Control u otros problemas misceláneos

Mientras está en el campo, es posible que necesite restablecer el OpenROV para remediar los errores extraños y aleatorios. Para este procedimiento se debe desenchufar el cable USB de la caja del PLC y volver a enchufarlo. Este método también funciona para otros problemas de control.

Pérdida aleatoria de energía de la batería

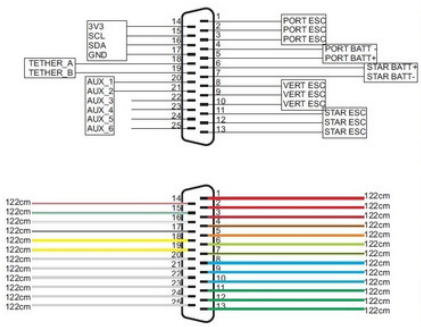
A menudo, cuando hay problemas de energía, primeramente habrá que asegúrese de que todas las baterías estén en contacto entre sí y, si corresponde, el adaptador de la batería y los terminales positivo y negativo en el tubo de la batería.

Verificar el voltaje a través de los pines correspondientes del conector DB-25. Para las baterías recomendadas, esto debería ser alrededor de 9V. Se comprobara la batería positivo y negativo de los cables: Naranja / Negro-Naranja y Verde claro / Negro-Verde claro. Un pequeño trozo de alambre puede ayudar a establecer una buena conexión con el polímetro.

Además, puede verificar cada batería individual para asegurarse de que tenga el voltaje correcto.

Hay que asegurarse de mover los cables alrededor o sacudir los tubos de la batería para identificar cualquier discontinuidad.

Se puede quitar la envoltura de plástico en la parte inferior de las baterías para ayudar en un buen contacto.



PIN	FUNCTION	COLOR	LENGTH
1	PORT ESC	RED	122CM
2	PORT ESC	RED	122CM
3	PORT ESC	RED	122CM
4	PORT BATTERY -	BLACK/ORANGE	122CM
5	PORT BATTERY +	ORANGE	122CM
6	STARBOARD BATT +	LIGHT GREEN	122CM
7	STARBOARD BATT -	BLACK/LIGHT GREEN	122CM
8	VERT ESC	BLUE	122CM
9	VERT ESC	BLUE	122CM
10	VERT ESC	BLUE	122CM
11	STARBOARD ESC	GREEN	122CM
12	STARBOARD ESC	GREEN	122CM
13	STARBOARD ESC	GREEN	122CM
14	3V3	GRAY/RED	122CM
15	SCL	GRAY/GREEN	122CM
16	SDA	GRAY/WHITE	122CM
17	GND	GRAY/BLACK	122CM
18	TETHER_A	YELLOW	122CM
19	TETHER_B	YELLOW	122CM
20	AUX_1	GRAY	122CM
21	AUX_2	GRAY	122CM
22	AUX_3	GRAY	122CM
23	AUX_4	GRAY	122CM
24	AUX_5	GRAY	122CM
25	AUX_6	GRAY	122CM

Figura 3.26: DB-25

Enredos

Los enredos es un problema muy comun que se encuentran en el este campo.

El mejor remedio es evitar la situación en absoluto. Esto se puede hacer evitando bucear en vegetación pesada.

A veces puede ser necesario tirar el OpenROV a la superficie con la correa. La correa es fuerte y puede manejar una gran cantidad de fuerza.

Hay que tener en cuenta que si se tira del OpenROV puede enredar el ROV, que es mucho más grande y difícil de desenredar. Si este es el caso, conduciremos el robot hasta nosotros o al menos hacia la superficie donde se puede recuperar. Luego, desconectaremos el cable USB del PLC y se tirará del ROV.

Siempre podemos cortar y volver a soldar la correa utilizando un tubo termorretráctil si fuese necesario.

Inundación del tubo principal

Este puede ser el mayor problema al que nos podemos enfrentar, por ello, lo primero que habría que realizar sería desconectar la alimentación del ROV (esto evitará que la electrólisis corroa los componentes electrónicos). Para hacer esto, desenchufamos el cable USB y retiraremos las baterías.

Si el agua que inundó el tubo no era agua dulce, enjuagamos los componentes electrónicos que se mojaron en agua dulce (desionizada¿destilada¿agua del grifo) para diluir y eliminar los minerales del área que se inundó.

Quitaremos los componentes electrónicos para exponer tantas superficies como sea posible.

Se enjuagará las áreas afectadas una vez más.

Secaremos bien cada una de las piezas de la electrónica. El aire comprimido puede ayudar.

Se colocaran todos los componentes electrónicos afectados en un lugar seco. Es útil tener una bolsa de llena de arroz o desecante.

Cuando vuelva a armar todo, inspeccionamos para detectar corrosión. Si vemos alguno, a menudo se puede quitar con un cepillo de dientes. La corrosión no significa necesariamente un fallo, pero es bueno eliminar lo que se pueda.

Capítulo 4

Integración de ROS en OpenROV

En este capítulo se va a describir como se ha realizado la integración de ROS con OpenROV.

4.1. Configuración del ROV

La electrónica del ROV consiste en un Arduino conectado a un Beagle Bone, la cual, ejecuta el software nativo OpenROV, que utiliza node.js y se comunica mediante mensajes JSON.

Como la BeagleBone tiene una computación limitada, y como el robot esta conectado al portátil, se decide implementar ROS en el portátil en vez de en el ROV. Con esta configuración, el ROS Master se ejecuta en el portátil e interactúa con los nodos de ROS que tengamos instalados (Rosbridge y gscam). Mientras, el ROV contendrá un nodo adicional (Roslibjs) que interactuará con Rosbridge para la comunicación de datos.

Roslibjs es la biblioteca principal de JavaScript para interactuar con ROS. Utiliza WebSocket para coectarse con Rosbridge y proporciona publicación, suscripción, llamadas de servicio, TF...

Rosbridge proporciona una API que puede traducir mensajes JSON en mensajes ROS. Actúa como un Nodo ROS que puede publicar o suscribirse a mensajes. Nos comunicamos con él a través de un conexión Websocket y Roslibjs.

Roslibjs interactúa con JavaScript para publicar o suscribirse a los mensajes de ROS. Utiliza Websockets para comunicarse con Rosbridge.

La figura a continuación muestra la estructura básica de ROS para la integración de OpenROV.

La configuración del ROV implica la instalación de las dependencias para el complemento OpenROV. La principal dependencia para el plugin de ROV es roslibjs, sin embargo roslibjs también tiene varias dependencias.

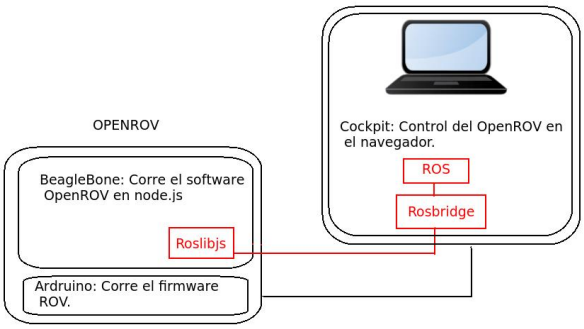


Figura 4.1: Diagrama de la conexión ROS y OpenROV

Para empezar configuraremos la BeagleBone a través de una conexión SSH. Para ello necesitamos configurar el hardware del ROV de la siguiente manera:

1. Con un cable Ethernet, conectaremos el BeagleBone al router.

2. Utilizarmeos un cable mini USB para conectar el BeagleBone al portátil, de esta forma, activaremos el hardware.

3. Haremos un ssh hacia la BeagleBone,

4.1: SSH

```
$ ssh rov@192.168.7.2
```

cuando pida la contraseña, deberemos introducir “OpenROV”.

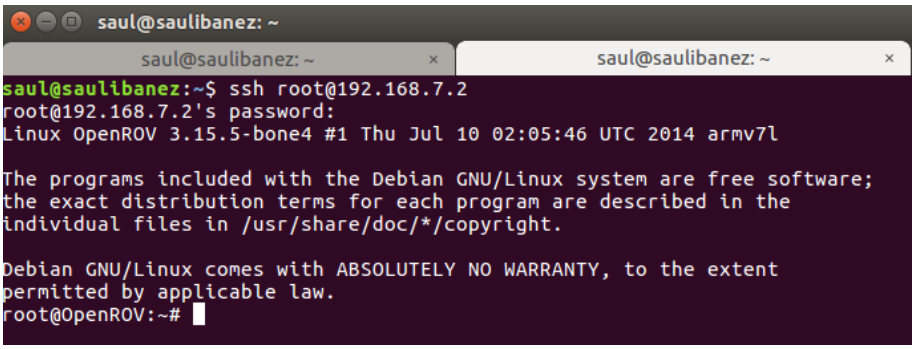


Figura 4.2: SSH

Instalaremos las dependencias de Roslibjs, para ello ejecutaremos los siguientes comandos:

4.2: Dependencias Roslibjs

```
sudo apt-get update
```

```
sudo apt-get install libcairo2-dev libjpeg-dev libpango1.0-dev libgif-dev
    build-essential g++
```

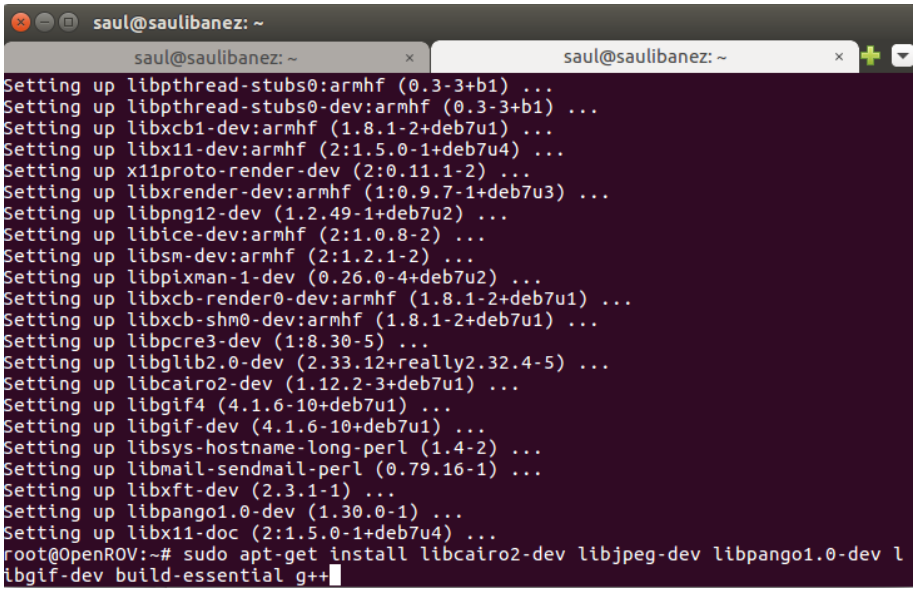


Figura 4.3: Dependencias Roslibjs

Seguidamente instalaremos otras dependencias (npm) de Roslibjs. Para que no surjan problemas(errores del proxy), deberemos deshabilitar el proxy y realizar un reseteo del sistema.

4.3: Dependencias Roslibjs

```
sudo /etc/init.d/openrov-proxy stop
sudo /etc/init.d/openrov restart
sudo chown -R $USER /usr/local
npm install canvas
```

```
saul@saulibanez: ~
saul@saulibanez: ~        saul@saulibanez: ~
root@OpenROV:~# cd /opt/openrov/cockpit/src/plugins
root@OpenROV:/opt/openrov/cockpit/src/plugins# npm install canvas
npm http GET https://registry.npmjs.org/canvas
npm http 200 https://registry.npmjs.org/canvas
npm http GET https://registry.npmjs.org/canvas/-/canvas-1.6.9.tgz
npm http 200 https://registry.npmjs.org/canvas/-/canvas-1.6.9.tgz
npm http GET https://registry.npmjs.org/nan
npm http 200 https://registry.npmjs.org/nan
npm http GET https://registry.npmjs.org/nan/-/nan-2.8.0.tgz
npm http 200 https://registry.npmjs.org/nan/-/nan-2.8.0.tgz

> canvas@1.6.9 install /opt/openrov/cockpit/node_modules/canvas
> node-gyp rebuild

make: Entering directory `/opt/openrov/cockpit/node_modules/canvas/build'
  SOLINK_MODULE(target) Release/obj.target/canvas-postbuild.node
  SOLINK_MODULE(target) Release/obj.target/canvas-postbuild.node: Finished
  COPY Release/canvas-postbuild.node
  CXX(target) Release/obj.target/canvas/src/Canvas.o
In file included from ../node_modules/nan/nan.h:199:0,
                 from ../src/Canvas.h:22,
                 from ../src/Canvas.cc:7:
../node_modules/nan/nan_maybe_pre_43_inl.h: In function 'Nan::Maybe<bool> Nan::D
efineOwnProperty(v8::Handle<v8::Object>, v8::Handle<v8::String>, v8::Handle<v8::
Value>, v8::PropertyAttribute)':
../node_modules/nan/nan_maybe_pre_43_inl.h:159:59: warning: suggest parentheses
around '&&' within '||' [-Wparentheses]
  CXX(target) Release/obj.target/canvas/src/CanvasGradient.o
In file included from ../node_modules/nan/nan.h:199:0,
                 from ../src/Canvas.h:22,
                 from ../src/CanvasGradient.cc:9:
../node_modules/nan/nan_maybe_pre_43_inl.h: In function 'Nan::Maybe<bool> Nan::D
efineOwnProperty(v8::Handle<v8::Object>, v8::Handle<v8::String>, v8::Handle<v8::
Value>, v8::PropertyAttribute)':
../node_modules/nan/nan_maybe_pre_43_inl.h:159:59: warning: suggest parentheses
around '&&' within '||' [-Wparentheses]
  CXX(target) Release/obj.target/canvas/src/CanvasPattern.o
In file included from ../node_modules/nan/nan.h:199:0,
                 from ../src/Canvas.h:22,
                 from ../src/CanvasPattern.cc:8:
```

Figura 4.4: Canvas

La última dependencia para el complemento es roslibjs, lo instalaremos usando: npm install roslib

Dentro de la carpeta /opt/openrov/cockpit/src/plugins se creará una carpeta llamada ros (sudo mkdir ros) y en ella, crearemos un fichero index.js en el cual se ejecutará la conexión con el ROS Master.

4.4: Conexión con ROS

```
function ros(name, deps) {
      console.log("ROS plugin started");

      // Comienza la sesion de ROS
      var ROSLIB = require("roslib")
      var ros = new ROSLIB.Ros({
            url : 'ws://192.168.7.1:9090'
      });

      ros.on('connection', function() {
            console.log('ROS connected to websocket');
      });
```

```
    ros.on('error', function(error) {
        console.log('ROS error connecting to websocket');
    });

     ros.on('close', function() {
        console.log('ROS closed websocket connection');
    });
};

module.exports = ros;
```

También se debe modificar el fichero hosts (sudo vim /etc/hosts) y en el realizaremos los siguientes cambios:

4.5: /etc/hosts

```
#127.0.1.1 OpenROV
192.168.7.2 OpenROV
192.168.7.1 saul.ibanez.local
```

4.2. Configuración de ROS

Antes de empezar cualquier conexión con ROS, deberemos instalarlo, en este punto, se ha decido instalar la distribucion de ROS Kinetic.

4.6: Instalacion de ROS

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc)
    main" > /etc/apt/sources.list.d/ros-latest.list'
sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key
    421C365BD9FF1F717815A3895523BAEEB01FA116
sudo apt-get update
sudo apt-get install ros-kinetic-desktop-full
apt-cache search ros-kinetic
sudo rosdep init
rosdep update
echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc
source ~/.bashrc
sudo apt-get install python-rosinstall python-rosinstall-generator
    python-wstool build-essential
```

Instalaremos Rosbridge

4.7: Rosbridge

```
sudo apt-get install ros-kinetic-rosbridge-server
```

Cuando se configuró ROS, creamos una carpeta llamada catkin_ws, en ella, se creó también la carpeta src, en la que crearemos el directorio openrov el que debe contener un fichero CmakeList.txt

El archivo CMakeLists.txt es la entrada al sistema de compilación CMake para crear paquetes de software. Cualquier paquete describe cómo crear el código y dónde instalarlo, el código será:

4.8: CMakeList.txt

```
cmake_minimum_required(VERSION 2.8.3)
project(openrov)

find_package(catkin REQUIRED COMPONENTS
   roscpp
   rospy
   std_msgs
   sensor_msgs
   geometry_msgs
   message_generation
)

add_message_files(
      FILES
      navdata.msg
      rovstatus.msg
      motortarget.msg
  )

add_service_files(
    FILES
)

generate_messages(
   DEPENDENCIES
      std_msgs
      sensor_msgs
      geometry_msgs
)

catkin_package(
      INCLUDE_DIRS include
      LIBRARIES openrov
      CATKIN_DEPENDS roscpp rospy message_runtime
                 std_msgs sensor_msgs geometry_msgs
      DEPENDS system_lib
)

include_directories(
   ${catkin_INCLUDE_DIRS}
)
```

Además, crearemos un directorio nombrado launch en el que se creará el fichero rosbridge.launch que contendrá:

4.9: websocket.launch

```
<launch>
   <include file="$(find
      rosbridge_server)/launch/rosbridge_websocket.launch"/>
</launch>
```

Este código conduce a la instalación de rosbridge_server e iniciará la conexión con ROV.

Para finalizar, y antes de ejecutar el .launch, deberemos ejecutar otro fichero, llamado set_remote_robot, el cual está configurado de la siguiente manera:

4.10: set_remote_robot

```
   export ROS_IP=192.168.7.1  #YOUR IP
   export ROS_HOSTNAME=saulibanez.local  #YOUR MACHINE
   export ROS_MASTER_URI=http://saulibanez.local:11311
```

4.3. Conexión de ROS en OpenROV

Habiendo realizado todos los pasos anteriores, ya estamos preparados para realizar la conexión de ROS en OpenROV.

Empezaremos conectando el OpenROV (solo con el cable MiniUSB en el portátil) abriremos un terminal y realizaremos un SSH a el BeagleBone (ssh rov@192.168.7.2).

Una vez conectados, desactivaremos el proxy (sudo /etc/init.d/openrov-proxy stop)y reinicamos elrobot (sudo /etc/init.d/openrov restart).

En otro terminal, ejecutaremos un roscore, que iniciará el ROS Master.

```
roscore http://saulibanez:11311/
saul@saulibanez: ~/catkin_ws        roscore http://saulibanez:11311/
saul@saulibanez:~$ roscore
... logging to /home/saul/.ros/log/14a24092-1b21-11e8-8f89-485ab613c113/roslaunc
h-saulibanez-32411.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://saulibanez:46558/
ros_comm version 1.12.12


SUMMARY
========

PARAMETERS
 * /rosdistro: kinetic
 * /rosversion: 1.12.12

NODES

auto-starting new master
process[master]: started with pid [32424]
ROS_MASTER_URI=http://saulibanez:11311/

setting /run_id to 14a24092-1b21-11e8-8f89-485ab613c113
process[rosout-1]: started with pid [32437]
started core service [/rosout]
```

Figura 4.5: Roscore

Teniendo conectado el ROS Master, podremos ejecutar el launch rosbridge.launch, que es el que iniciará el nodo de rosbridge para comunicarse con OpenROV.

Ejecutando el código:

4.11: launch

```
$ roslaunch openrov rosbridge.launch
```

```
/opt/ros/kinetic/share/rosbridge_server/launch/rosbridge_websocket.launch http://saulibanez.local:11311
saul@saulibanez: ...   /opt/ros/kinetic/s...   saul@saulibanez: ...   saul@saulibanez: ...   saul@saulibanez: ...
 * /rosdistro: kinetic
 * /rosversion: 1.12.12

NODES
  /
    rosapi (rosapi/rosapi_node)
    rosbridge_websocket (rosbridge_server/rosbridge_websocket)

auto-starting new master
process[master]: started with pid [5842]
ROS_MASTER_URI=http://saulibanez.local:11311

setting /run_id to d67ecea4-0697-11e8-abcb-485ab613c113
process[rosout-1]: started with pid [5855]
started core service [/rosout]
process[rosbridge_websocket-2]: started with pid [5865]
process[rosapi-3]: started with pid [5869]
registered capabilities (classes):
 - rosbridge_library.capabilities.call_service.CallService
 - rosbridge_library.capabilities.advertise.Advertise
 - rosbridge_library.capabilities.publish.Publish
 - rosbridge_library.capabilities.subscribe.Subscribe
 - <class 'rosbridge_library.capabilities.defragmentation.Defragment'>
 - rosbridge_library.capabilities.advertise_service.AdvertiseService
 - rosbridge_library.capabilities.service_response.ServiceResponse
 - rosbridge_library.capabilities.unadvertise_service.UnadvertiseService
[INFO] [1517410985.144924]: Rosbridge WebSocket server started on port 9090
[INFO] [1517413486.040718]: Client connected.  1 clients total.
[INFO] [1517413486.452515]: [Client 0] Subscribed to /openrov/cmd_rate
[INFO] [1517413486.456547]: [Client 0] Subscribed to /openrov/cmd_pose
[INFO] [1517413514.254572]: Client disconnected. 0 clients total.
[INFO] [1517413528.973119]: Client connected.  1 clients total.
[INFO] [1517413529.072633]: [Client 1] Subscribed to /openrov/cmd_rate
[INFO] [1517413529.078050]: [Client 1] Subscribed to /openrov/cmd_pose
```

Figura 4.6: Launch

También se comprobará en el terminal de OpenROV que la conexión se ha realizado

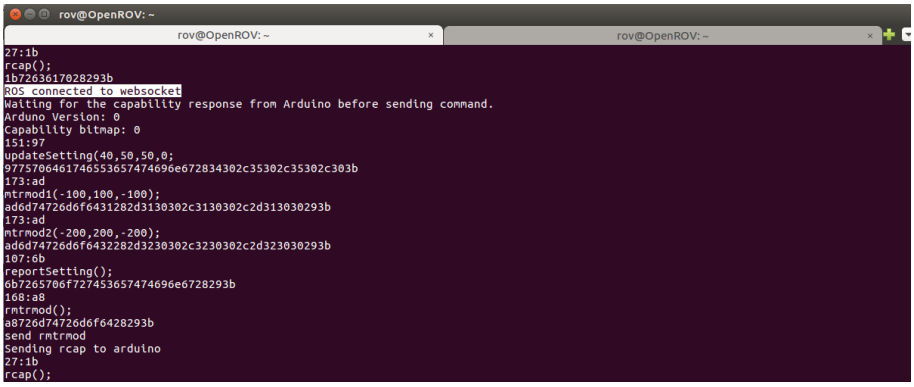


Figura 4.7: Conexión en ROV

Para asegurarse de que todo esté funcionando, abrimos otra pestaña en el terminal e ingresaremos:

4.12: rostopic list

```
$ rostopic list
```

Esto mostrará una lista de los temas que están siendo publicados por el ROV, así como algunos temas predeterminados.

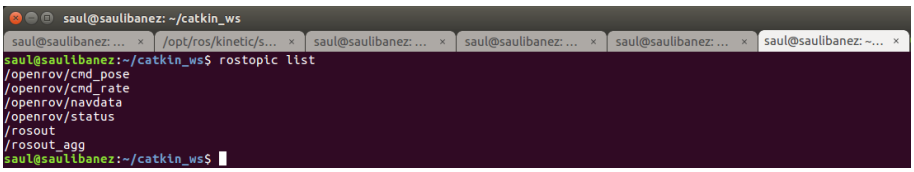


Figura 4.8: rostopic list

Para ver información sobre el nodo en particular podremos ejecutar el comando :rostopic info.

4.13: rostopic info

```
$ rostopic info
```

```
saul@saulibanez: ~/catkin_ws
saul@saulibane... | /opt/ros/kinetic... | saul@saulibane... | saul@saulibane... | saul@saulibane... | saul@saulibanez...
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/cmd_rate
Type: geometry_msgs/Twist

Publishers: None

Subscribers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)


saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata /
Usage: rostopic info /topic

rostopic: error: you may only specify one topic name
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata
Type: openrov/navdata

Publishers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)

Subscribers: None


saul@saulibanez:~/catkin_ws$ rosm
rosmake        rosmaster      rosmsg         rosmsg-proto
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata
Type: openrov/navdata

Publishers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)

Subscribers: None


saul@saulibanez:~/catkin_ws$ rosmsg show /openrov/navdata
Unable to load msg [/openrov/navdata]: invalid name [/openrov/navdata]
```

Figura 4.9: rostopic info (1)

```
saul@saulibanez: ~/catkin_ws
saul@saulibane... | /opt/ros/kinetic... | saul@saulibane... | saul@saulibane... | saul@saulibane... | saul@saulibanez...
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/cmd_pose
Type: geometry_msgs/Pose

Publishers: None

Subscribers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)


saul@saulibanez:~/catkin_ws$ rostopic info /openrov/cmd_rate
Type: geometry_msgs/Twist

Publishers: None

Subscribers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)


saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata /
Usage: rostopic info /topic

rostopic: error: you may only specify one topic name
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata
Type: openrov/navdata

Publishers:
 * /rosbridge_websocket (http://saulibanez.local:38123/)

Subscribers: None


saul@saulibanez:~/catkin_ws$ rosm
rosmake        rosmaster      rosmsg         rosmsg-proto
saul@saulibanez:~/catkin_ws$ rostopic info /openrov/navdata
Type: openrov/navdata
```

Figura 4.10: rostopic info (2)

Capítulo 5

Pruebas de Campo

Se adjunta los videos de motores y prueba de estanqueidad.

En este video se ve como funcionan los motores:

motores en funcionamiento

Prueba de estanqueidad en la bañera:

estanqueidad en la bañera

Capítulo 6

Conclusiones

Los objetivos de este trabajo fueron dos: por un lado, presentar a los robots submarinos, en particular, OpenROV y por otro lado, la integración de ROS, el framework de robótica más extendido y utilizado con la tecnología del ROV.

Los robots submarinos han cambiado notablemente la exploracion del fondo marino. Además, estos robots han permitido realizar intervenciones en desastres como en los casos de fugas en instalaciones petrolíferas. Sin duda, el desarrollo de esta tecnología es importante debido a las necesidades actuales y a las posibles futuras aplicaciones de estos robots.

Ha sido un trabajo entretenido ya que se ha tenido que realizar todo desde cero, he podido aprender sobre la tecnología de ROV y explorar nodos de ROS.

Los siguientes pasos de este entretenido TFG es intentar comprobar si la leyenda de Nessie es cierta.

Bibliografía

[1] B. Adams, "Ros integration with openrov," http://blog.openrov.com/ros-integration-with-openrov/, 2015.

[2] Anónimo, "Una introducción a los robots móviles," http://www.aadeca.org/pdf/CP_monografias/monografia_robot_movil.pdf, 2008.

[3] W. Garage, "Ros," http://www.ros.org/, 2010.

[4] W. W. Royce, "Modelo en cascada," https://es.wikipedia.org/wiki/Desarrollo_en_cascada, 1970.

[5] E. Stackpole and D. Lang, "Openrov," https://openrov.dozuki.com/ , and https://openrov.com, 2015.

[6] D. D. Tarascó, "Desarrollo de sistema de telemetría para teleoperación de robot submarino," Proyecto Fin de Carrera, Universidad Simón Bolívar, 2016.